digital

TCP/IP Networking on OpenVMS Systems

OpenVMS

TCP/IP Networking on OpenVMS Systems

Order Number: AA–QJGDA–TE

May 1995

This manual provides an introductory overview of TCP/IP networking and internets, and describes OpenVMS DCL support for TCP/IP capabilities.

Revision/Update Information: This is a new manual.
Software Version: OpenVMS Alpha Version 6.2
OpenVMS VAX Version 6.2

Digital Equipment Corporation
Maynard, Massachusetts

May 1995

Digital Equipment Corporation makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patent rights, nor do the descriptions contained in this publication imply the granting of licenses to make, use, or sell equipment or software in accordance with the description.

Possession, use, or copying of the software described in this publication is authorized only pursuant to a valid written license from Digital or an authorized sublicensor.

Digital conducts its business in a manner that conserves the environment and protects the safety and health of its employees, customers, and the community.

The following are trademarks of Digital Equipment Corporation: AXP, Bookreader, DECnet, DECwindows, Digital, DNA, OpenVMS, OpenVMS RMS, PATHWORKS, ULTRIX, VAX, VAX DOCUMENT, VMScluster, VT100, VT300, and the DIGITAL logo.

The following are third-party trademarks:

Adobe, Display PostScript, and PostScript are registered trademarks of Adobe Systems Inc.

AppleTalk is a registered trademark of Apple Computer, Inc.

BSD is a trademark of the University of California, Berkeley, CA.

Ethernet is a registered trademark of Xerox Corporation.

IBM is a registered trademark of International Business Machines Corporation.

IEEE is a registered trademark of the Institute of Electrical and Electronics Engineers.

Internet is a registered trademark of Internet, Inc.

MultiNet is a registered trademark of TGV, Inc.

NetWare is a registered trademark of Novell, Inc.

NFS is a registered trademark of Sun Microsystems, Inc.

OSF and OSF/1 are registered trademarks of the Open Software Foundation, Inc.

PathWay is a trademark of The Wollongong Group, Inc.

TCPware is a registered trademark of Process Software Corporation.

UNIX is a registered trademark of Unix Systems Laboratories, Inc., a wholly owned subsidiary of Novell, Inc.

All other trademarks and registered trademarks are the property of their respective holders.

ZK6436

This document is available on CD–ROM.

This document was prepared using VAX DOCUMENT Version 2.1.

Send Us Your Comments

We welcome your comments on this or any other OpenVMS manual. If you have suggestions for improving a particular section or find any errors, please indicate the title, order number, chapter, section, and page number (if available). We also welcome more general comments. Your input is valuable in improving future releases of our documentation.

You can send comments to us in the following ways:

- Internet electronic mail: `openvmsdoc@zko.mts.dec.com`
- Fax: `603-881-0120 Attn: OpenVMS Documentation, ZKO3-4/U08`
- Online form

  Print or edit the online form SYS$HELP:OPENVMSDOC_COMMENTS.TXT. Send the completed online form by electronic mail to our Internet address, or send the completed hardcopy form by fax or through the postal service.

  Please send letters or the form to:

  ```
  Digital Equipment Corporation
  Information Design and Consulting
  OpenVMS Documentation
  110 Spit Brook Road, ZKO3-4/U08
  Nashua, NH  03062-2698
  USA
  ```

Thank you.

How To Order Additional Documentation

Use the following table to order additional documentation or information. If you need help deciding which documentation best meets your needs, call 800-DIGITAL (800-344-4825).

Telephone and Direct Mail Orders

Location	Call	Fax	Write
U.S.A.	DECdirect 800.DIGITAL 800.344.4825	Fax: 800.234.2298	Digital Equipment Corporation P.O. Box CS2008 Nashua, NH 03061
Puerto Rico	809.781.0505	Fax: 809.749.8300	Digital Equipment Caribbean, Inc. 3 Digital Plaza, 1st Street, Suite 200 P.O. Box 11038 Metro Office Park San Juan, Puerto Rico 00910–2138
Canada	800.267.6215	Fax: 613.592.1946	Digital Equipment of Canada, Ltd. Box 13000 100 Herzberg Road Kanata, Ontario, Canada K2K 2A6 Attn: DECdirect Sales
International	—	—	Local Digital subsidiary or approved distributor
Internal Orders	DTN: 264.3030 603.884.3030	Fax: 603.884.3960	U.S. Software Supply Business Digital Equipment Corporation 10 Cotton Road Nashua, NH 03063–1260

ZK–7654A–GE

Contents

4 Mapping UNIX to OpenVMS Identification Code

5 Commands Common to All TCP/IP Products That Run on OpenVMS

A OpenVMS TCP/IP Software Vendors

B Using DECnet Over TCP/IP

Index

Figures

Tables

Preface

Intended Audience

This manual is intended for anyone who is interested in using TCP/IP networking on either the OpenVMS Alpha or the OpenVMS VAX operating system.

Readers may be new to networking or may be familiar with the traditional DECnet networking interface on OpenVMS systems.

Document Structure

This manual contains the following chapters:

- Chapter 1 introduces TCP/IP networking, internetworking, and the Internet.
- Chapter 2 summarizes TCP/IP networking architecture, layers, protocols, and addressing.
- Chapter 3 describes common TCP/IP applications that perform general network operations. The chapter also describes Internet information retrieval tools.
- Chapter 4 discusses mapping UNIX identification codes to OpenVMS user names in TCP/IP applications.
- Chapter 5 specifies the reference format of OpenVMS DCL commands that can be used to invoke TCP/IP capabilities provided by layered product software running on OpenVMS systems.
- Appendix A lists vendors and the software products they supply for TCP/IP networking services to OpenVMS systems. These products run as layered software on the OpenVMS operating system.
- Appendix B summarizes DECnet/OSI.

Related Documents

Refer to the following documents for more information about TCP/IP software features supported by OpenVMS. To order these documents, see the *Overview of OpenVMS Documentation* or contact your Digital representative.

- *OpenVMS DCL Dictionary*
- *OpenVMS Version 6.2 Release Notes*

See the appropriate vendor documentation for information about each of the layered TCP/IP software products that run on OpenVMS systems. These TCP/IP products are listed in Appendix A.

Conventions

The name of the OpenVMS AXP operating system has been changed to OpenVMS Alpha. Any references to OpenVMS AXP or AXP are synonymous with OpenVMS Alpha or Alpha.

The following conventions are used to identify information specific to OpenVMS Alpha or to OpenVMS VAX:

Alpha	The Alpha icon denotes the beginning of information specific to OpenVMS Alpha.
VAX	The VAX icon denotes the beginning of information specific to OpenVMS VAX.
◆	The diamond symbol denotes the end of a section of information specific to OpenVMS Alpha or to OpenVMS VAX.

In this manual, every use of DECwindows and DECwindows Motif refers to DECwindows Motif for OpenVMS software.

The following conventions are also used in this manual:

( )	In command format descriptions, parentheses indicate that, if you choose more than one option, you must enclose the choices in parentheses.
[ ]	In command format descriptions, brackets indicate optional elements. You can choose one, none, or all of the options. (Brackets are not optional, however, in the syntax of a directory name in an OpenVMS file specification or in the syntax of a substring specification in an assignment statement.)
{ }	In command format descriptions, braces indicate a required choice of options; you must choose one of the options listed.
boldface text	Boldface text represents the introduction of a new term or the name of an argument, an attribute, or a reason. Boldface text is also used to show user input in Bookreader versions of the manual.
italic text	Italic text indicates important information, complete titles of manuals, or variables. Variables include information that varies in system output (Internal error *number*), in command lines (/PRODUCER=*name*), and in command parameters in text (where *device-name* contains up to five alphanumeric characters).
UPPERCASE TEXT	Uppercase text indicates a command, the name of a routine, the name of a file, or the abbreviation for a system privilege.
`struct`	Monospace type in text identifies the following C programming language elements: keywords, the names of independently compiled external functions and files, syntax summaries, and references to variables or identifiers introduced in an example.
-	A hyphen in code examples indicates that additional arguments to the request are provided on the line that follows.
numbers	All numbers in text are assumed to be decimal unless otherwise noted. Nondecimal radixes—binary, octal, or hexadecimal—are explicitly indicated.

1

Introduction to TCP/IP Networking

An increasingly important aspect of using computers is the capability to communicate over open networks. Computer systems of similar or different design should be able to communicate with each other. In addition, various physical networks must be interconnected to form large networks called **internets**.

This chapter describes how Transmission Control Protocol/Internet Protocol (TCP/IP) networking meets these needs and how OpenVMS systems support TCP/IP networking. The chapter also describes how TCP/IP internetworking capabilities have made possible the global Internet, an openly accessible, worldwide research and commercial network.

1.1 Overview

This section introduces the primary concepts and features that characterize TCP/IP networking and summarizes networking capabilities available on OpenVMS systems.

1.1.1 What Is TCP/IP Networking?

A network consists of two or more computer systems linked by communications software and hardware for the purpose of exchanging information and sharing resources. Data originated on one system can be routed through the network until it reaches its destination on another system. The design of communications software that supports the flow of information is based on networking **protocols**: services and rules for exchanging information between systems.

One of the most widely used networking protocols is TCP/IP. It is a common set, or "suite," of protocols that work together to provide the services necessary to interconnect computer systems and to interconnect networks. TCP/IP was designed to permit connection of computer systems of dissimilar design. TCP/IP application programs allow users to interact with remote processors. TCP/IP can also be used to communicate across many interconnected networks.

The main characteristics of TCP/IP are openness, interoperability, and distributed client/server design.

1.1.1.1 Openness

The TCP/IP suite of protocols implements open networking standards that support open system interconnection. An open system is one for which the specifications are available to the public. Open specifications do not rely on a particular technology or product. They allow users to determine what open systems and other capabilities the user needs. TCP/IP technology, which accommodates a variety of underlying network technologies, permits connection of multiple kinds of computers in an open network environment. See Section 2.1 for a comparison of TCP/IP architecture with the Open Systems Interconnection (OSI) model.

1.1.1.2 Interoperability

In a TCP/IP networking environment multiple systems from various vendors can work together, share data, and integrate applications. A TCP/IP internet is useful for running application programs that carry out tasks such as accessing remote resources. These applications can interoperate with different applications running on other systems that support TCP/IP standards. A user does not need to know about TCP/IP software or data paths in order to run the applications. See Chapter 3 for information about commonly used TCP/IP applications.

1.1.1.3 Distributed Client/Server Design

TCP/IP software supports the use of client/server configurations in a distributed networking environment. A distributed system stores data and information on many computers, instead of on just one computer. In the client/server model, two software programs running on separate computers work together: one program, called the **client**, makes use of resources supplied by the other program, called the **server**. A server can receive a request from a client anywhere in a TCP/IP network, accept the request, and return the results or data to the client. Server programs are application-level programs that can execute on one or more machines of any size, including PCs, in a TCP/IP network.

One example of a client/server model is a bank clerk who uses client software running on a PC to enter deposit information about a customer's account. The server software, running on the computer at the bank's main branch, processes the request about the customer account and returns the customer's balance to the client software. Chapter 3 describes commonly used TCP/IP applications that implement the client/server model.

1.1.2 Communicating Over TCP/IP Networks

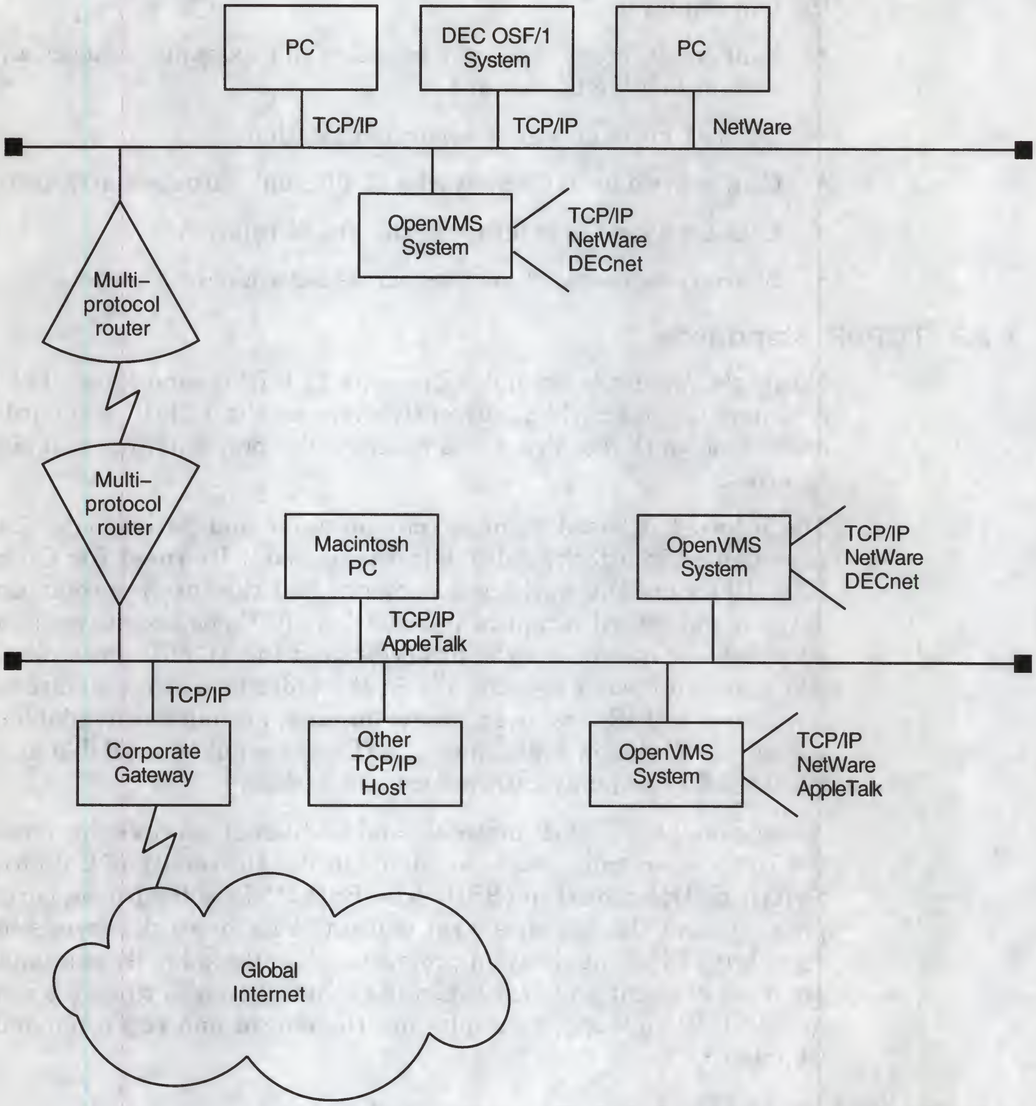
Each end system connected to a TCP/IP network is called a **host**. Each host has a unique name and address. The local host is the system you are using, and the remote host is the system with which you are communicating. Hosts are connected by **lines** that carry information between the hosts. The line is the physical path over which data can pass from one host to another. (Examples of lines are telephone lines, fiber-optic cables, and satellites.)

A TCP/IP network is called a packet-switching network. Information is transmitted in small packets of data rather than as a continuous stream from host to host. For example, a file to be transmitted from one host to another is divided into many small packets that are sent across the network one at a time. Each packet contains information about the address of the destination host. At the destination, the packets are reassembled.

The packets that comprise the network traffic are combined (multiplexed) onto high-capacity machine interconnections for transmission across the network or internet. Because packets from different sources are mixed together, many users can use the same line simultaneously. Individual packets can take different paths to the destination. The basic unit of data transmitted by TCP/IP is called a **datagram**.

The process of directing a data message from a source host to a destination host is called **routing**. For hosts not directly connected to each other, data can be forwarded from the source to the destination through intervening hosts.

Figure 1–2 Multiprotocol Network Topology



ZK–7438A–GE

1.2.2 Using TCP/IP for Internetworking

TCP/IP was developed as a result of research funded by the Defense Advanced Research Projects Agency (DARPA), an agency of the U.S. Department of Defense. The need to connect many computers with different hardware, operating systems, and networking technologies led to the development of the ARPANET, on which TCP/IP was first implemented. DARPA also made the TCP/IP implementation available to university researchers for use with UNIX operating systems.

The growing diversity of new networking technologies caused DARPA to study network connectivity, or internetworking. By 1983, the Department of Defense mandated that all computers connected to long-haul networks use TCP/IP. The success of the TCP/IP technology and internetworking has resulted in the enormous growth of the global Internet (see Section 1.3).

TCP/IP is widely used within organizations or industries to create internets that may or may not be connected to the global Internet. TCP/IP capabilities include the following:

- Connect different types of computers (for example, connect an OpenVMS system to a UNIX system)
- Connect hosts at widely separated locations
- Connect two or more networks of different hardware architecture
- Connect a host or internet to the global Internet
- Share resources such as files across networks or internets

1.2.3 TCP/IP Standards

No single vendor or organization owns TCP/IP technologies. The Internet Architecture Board (IAB) currently oversees the TCP/IP standards. This group assigns network addresses and specifies the protocols that can be used on the Internet.

The Internet is based on numerous protocols and conventions. Each protocol is explained in a technical publication called a **Request for Comment** or RFC. RFCs are the working documents that the Internet community uses to develop and record technical information. RFCs document work on the Internet, proposals for new or revised protocols, and the TCP/IP protocol standards. The RFCs are numbered sequentially in the order in which they are written. Each new or revised RFC is given a new number; an index is available to help identify the latest version of a document. RFCs are publicly available and are stored electronically at many Internet computer sites.

As an example, TCP/IP protocols and additional services are implemented in the UNIX operating system available in the University of California's **Berkeley Software Distribution** (BSD). The BSD UNIX software supports basic TCP/IP protocols and also supplies a set of utilities for network services in addition to the standard TCP/IP application programs. The Berkeley R commands and services are a set of client and server facilities that allow you to access remote resources in a TCP/IP network. Examples are the **rlogin** and **rcp** commands described in Chapter 3.

1.3 Global Connectivity Over the Internet

The Internet is a worldwide network encompassing tens of thousands of individual networks linked together with a high-speed backbone network. The **backbone** network comprises telephone and fiber-optic links, lasers, microwaves, satellites, and other equipment connecting networks and computers throughout the world. The backbone network evolved in part from the high-speed network developed by the National Science Foundation (NSF), a U.S. government agency, to connect their supercomputer sites and make them accessible to scientists, researchers, and engineers.

Examples of the many kinds of organizations linked through the Internet are scientific research agencies, government laboratories, universities and other educational facilities, libraries and archives, commercial corporations, and groups of people with common interests. Some commercial companies act as Internet service providers, making access to the Internet available to organizations and individuals.

A great variety of information is accessible through the Internet, ranging from scientific and academic material to commercial product documentation to news databases and forums for exchanging opinions and ideas.

The Internet provides global connectivity; major Internet facilities are located on every continent and in most countries around the world. In many countries, the Internet is readily accessible to individuals as well as regional and local groups. In addition, gateways permit access to resources on many non-Internet networks. The Internet is expanding at a very rapid rate in terms of numbers of connections and the amount of message traffic.

Internet connectivity enables users to communicate with other connected hosts in distant countries as though the hosts were connected to the same local network. The design of the Internet allows an Internet user to access and retrieve vast amounts of information from anywhere in the world.

2

How Does TCP/IP Work?

Two networking models support open systems interconnection. The first model, TCP/IP, is based on a suite of protocols in which each protocol solves a particular network communications problem. The second model, OSI, is based on international standards.

This chapter compares the models, and then provides an overview of TCP/IP architectural design, layers and protocols, and the TCP/IP naming mechanism.

2.1 Introduction to TCP/IP Architecture

Network software design is commonly based on a networking model made up of several layers that work together. Each **layer** is a group of related functions with its own characteristic protocols and purpose. The layers are built on top of one another so that each layer uses services provided by the layer beneath it. Information flows down through the layers of the sending host and up through the layers of the receiving host.

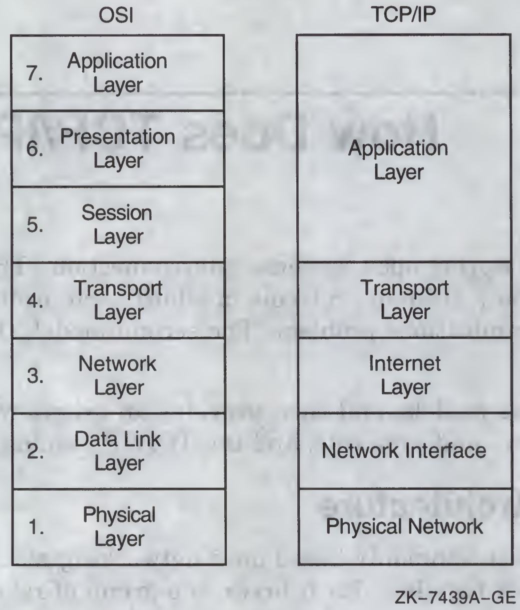
The architectural model on which networking implementations are based defines the ways in which operating systems can communicate with each other. Networking protocols, services, and interfaces allow systems that implement the model to communicate.

2.1.1 OSI and TCP/IP Models

The OSI model is a layered architecture that interconnects systems from different vendors in an open systems network. The OSI model is based on a set of international standards developed by the International Organization for Standardization (ISO). The seven layers of the OSI model are shown in Figure 2–1. The lower layers (1 through 4) provide for reliable transfer of information between two communicating systems. The upper layers (5 through 7) provide services that enable user applications to communicate with each other.

The TCP/IP model can be used in a heterogeneous environment that has equipment from many different vendors. Layers in the TCP/IP model are also shown in Figure 2–1.

Figure 2–1 Comparison of Layers in the OSI and TCP/IP Models



An OpenVMS system can support both TCP/IP and OSI capabilities on the same system. OSI protocols are incorporated in Digital's DECnet software (DECnet/OSI for OpenVMS), which can run concurrently with TCP/IP networking products on the same OpenVMS system.

2.1.2 Comparing the OSI Model with the TCP/IP Model

In comparing the TCP/IP model with the OSI model, TCP/IP can be viewed as supplying the functions of layers 3 and 4 of the OSI model. TCP/IP provides a number of protocols for the internet layer (corresponding to layer 3, the network layer, of the OSI model) and the transport layer (corresponding to layer 4, the transport layer, of the OSI model).

As shown in Figure 2–1, however, the TCP/IP model does not follow the OSI model exactly for all layers. For the upper-level layers, TCP/IP applications provide the services of the presentation and session layers of the OSI model. In addition, TCP/IP does not provide specific protocols for the bottom two layers that correspond to the physical layer and the data link layer of the OSI model. Instead, TCP/IP interfaces with whatever protocols are available for the physical network and the network interface.

Implementation of the OSI model places emphasis on providing a reliable data transfer service, while the TCP/IP model treats reliability as an end-to-end problem. Each layer of the OSI model detects and handles errors; all data transmitted includes checksums. The transport layer of the OSI model checks source-to-destination reliability.

In the TCP/IP model, reliability control is concentrated at the transport layer. The transport layer handles all error detection and recovery. Individual hosts or links can lose data without making any attempt at recovery. Corrupted datagrams can be discarded at internal gateways, and datagrams can be rerouted or dropped if network line problems occur. The TCP/IP transport layer uses checksums, acknowledgments, and timeouts to control transmissions and provides end-to-end verification.

Another contrast between the OSI model and the TCP/IP model is the role of the host system. Hosts on OSI implementations do not handle network operations, but TCP/IP hosts participate in most network protocols. TCP/IP hosts carry out such functions as end-to-end verification, routing, and network control. The TCP/IP internet can be viewed as a data stream delivery system involving intelligent hosts.

The following sections describe TCP/IP networking design, including the functions of the different layers and protocols.

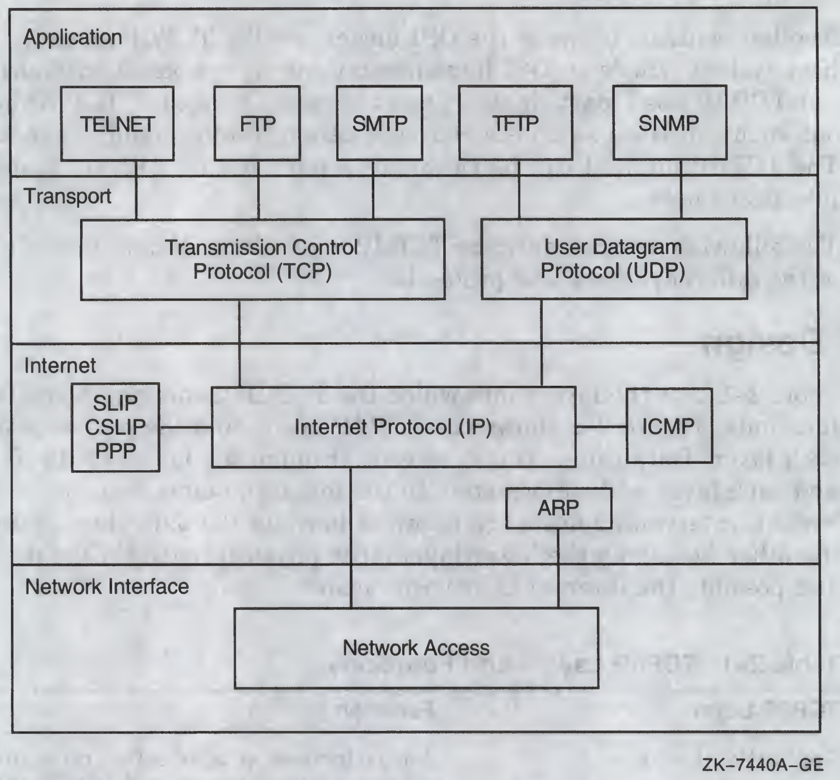
2.2 TCP/IP Design

Table 2–1 lists the layers into which the TCP/IP model organizes TCP/IP software functions. Figure 2–2 shows the TCP/IP layers and the protocols supported at each layer. Data can pass successively through the layers of the TCP/IP software and each layer adds information to the message being transmitted to the remote host. If intervening hosts are required to route the data through the network to the other host, only the lower layers (the physical network, the network interface, and possibly the internet layer) are involved.

Table 2–1 TCP/IP Layers and Functions

TCP/IP Layer	Function
Application Layer	A user invokes an application program that accesses a service available across a TCP/IP internet. The application passes data to and receives data from the transport layer. Protocols available at the application layer are described in Section 2.3.
Transport Layer	This layer provides services that permit an application program on one host to communicate with an application program on a remote host. The transport layer divides the stream of data into packets, adds a destination address, and passes the packets to the next layer. The transport uses two protocols, TCP and UDP, described in Section 2.3.
Internet Layer	This layer ensures that data are routed to the correct destination. The internet layer encapsulates the packet received from the transport layer into a datagram, adds a header, and determines the routing requirement. For incoming datagrams, it determines which transport protocol should handle the packet. The internet layer uses the Internet Protocol (IP), as described in Section 2.4.
Network Interface	This layer controls access to network transmission mechanisms. The network interface is responsible for accepting IP datagrams and transmitting them over a specific network. The interface can be a device driver (connected to a LAN) or a subsystem with its own data link protocol.
Physical Network	The hardware connection provides the physical interconnection between the host and the network.

Figure 2–2 TCP/IP Layers and Protocols



2.3 Transport Layer

The transport layer provides for transport of data. A user application is designed to choose between the following two transport layer protocols, depending on the services the application needs:

- Transmission Control Protocol (TCP) for connection-oriented, reliable end-to-end data transfer. TCP provides data streams.
- User Datagram Protocol (UDP) for connectionless, unreliable data transfer, involving simple transfer of data without acknowledgment or flow control. UDP provides datagrams.

2.3.1 Transmission Control Protocol

TCP is used by applications that require reliable data transmission between internet hosts. TCP provides a highly reliable data stream between transport layers on different hosts.

TCP supplies the following services in carrying out end-to-end verification between the source and destination for detection and recovery of lost datagrams:

- Requires acknowledgment from the receiving transport layer within a specific time
- Provides a sequence number to ensure that data are delivered in the order in which they were sent
- Retransmits lost data
- Transmits and verifies checksums

- Performs flow control to specify how much data can be accepted before an acknowledgment is sent

Before the transmission of data, TCP establishes a connection between the two transport layers through the exchange of messages. Because multiple application programs may run on a single host, TCP uses **protocol port** numbers to distinguish among the multiple destinations within a given host computer. A particular port on a host can be addressed by a client to request a particular service. To identify a connection, TCP specifies a pair of endpoints, defining each endpoint as consisting of a host IP address and a TCP port on that host.

The following application protocols use TCP:

- TELNET, for remote login to other hosts in the network
- FTP, the File Transfer Protocol, for file transfer
- SMTP, Simple Mail Transfer Protocol, for electronic mail

TELNET and FTP applications are described in Chapter 3.

2.3.2 User Datagram Protocol

UDP is an alternative transport mechanism that is more efficient than TCP but does not guarantee that the data will be delivered. Discovering that a particular packet, or datagram, was delivered out of sequence or not delivered is up to the user of UDP.

UDP permits an application to send a message to a destination application on another network without requiring that the destination application be active when the message is sent. UDP sends datagrams but does not guarantee delivery or protection against duplicate datagrams. The message delivery is connectionless: UDP considers the delivery complete once the message is placed on the network.

The following protocols can use UDP:

- BOOTP, the protocol for network booting
- TFTP, the Trivial File Transfer Protocol, for downloading files
- SNMP, the Simple Network Management Protocol, for network management
- NFS, the network file system

2.4 Internet Layer

The TCP/IP internet layer moves data from one host to another even if the hosts are on different networks. The internet layers on cooperating hosts are in communication with each other while they move data across a network. Each intervening host along the logical communications path moves the data closer to the destination host.

The primary protocol used to move data is the Internet Protocol (IP), which provides the following services:

- Addressing: Determining the route to deliver data to the destination host
- Fragmentation: Breaking the message into pieces if an intervening network cannot handle a large message

IP provides a connectionless method of delivering data from one host to another. It does not guarantee delivery and does not provide sequencing of datagrams (packaged in packets). IP attaches a header to the datagram that includes the source address and the destination address, both of which are unique internet addresses. If an intervening router needs to change the size of a message so a router can handle the message, IP fragments the message into smaller packets that can be reassembled at the destination host.

2.4.1 IP Routing

IP routing involves using the destination IP address of a datagram to decide where to send the datagram. IP routing passes to the network interface software the IP address of the next machine to which the datagram should be routed. The datagram is encapsulated, the IP address is mapped to a physical address, and the frame is transmitted to the next host.

If the source and destination hosts are in the same network, the routing is direct. If the datagram must be sent through a gateway to another network, the routing is indirect. The host sends an indirectly routed datagram to the nearest gateway; the datagram is then routed from gateway to gateway until it can be routed directly across a network to the destination host.

2.4.2 IP Addressing

Each host in a network has a unique IP address that is used in communicating with the host. Addresses, which are assigned by a central agency, are in 32-bit binary format, usually expressed as 8-bit fields separated by decimal points. Each field can have a value from 0 to 255 (for example, 97.0.5.110). IP addresses are divided into the following classes:

- Class A addresses are for networks with an extremely large number of hosts. The first byte is the network number; the last three bytes are the host number.
- Class B addresses are for intermediate-sized networks. The first two bytes are the network number; the last two bytes are the host number.
- Class C addresses are for small networks of fewer than 256 hosts. The first three bytes are the network number; the last byte is the host number.
- Class D addresses are multicast addresses.

A **subnet** is a network within a network. Organizations can use subnet addressing to divide an assigned network. For example, an organization that uses a Class B network number, might choose to subnet the network to effectively have more than 250 networks by using 8 bits of subnet. The Class B network number becomes a Class C network number. Subnets can be used to add hosts without disrupting the rest of the network, especially if the network contains a number of gateways.

2.4.3 Internet Host Names

Each host computing system in a TCP/IP network or internet is identified by a unique host name as well as a unique IP host address. TCP/IP supplies a mechanism for translating the host name to the host address that is required by the IP protocol.

The domain name system (DNS) is one example of a distributed name/address mechanism used in the global Internet. It provides for a hierarchy of host names and distributes host name and address information throughout the Internet. Another example is the BIND Resolver.

The name space for Internet hosts supports hierarchically arranged host names, called **domain** names. The domain name uniquely identifies a host computer that is connected to the Internet. The top-level domain name in the hierarchy can represent an organizational domain or a geographical domain. In the United States, examples of typical organizational domain names are *com* (for commercial organizations) and *edu* (for educational institutions). Internationally, the geographical domain name is a standard two-letter international country abbreviation (such as *au* for Australia and *fr* for France).

The top-level domain name can be divided into subdomain names that further identify the host. The subdomain names are arranged to the left of the top-level domain name and are separated by periods. For example, *computer-name.company.com* is the format.

The Internet address of a user who is logged in to an Internet host is in the form *userid@domain* in which *userid* is the user's login name.

2.4.4 Other Internet Layer Protocols

Protocols that interact with the Internet Protocol provide services that manage data movement problems. These protocols include:

- ARP, the Address Resolution Protocol, translates software addresses to hardware addresses for use by the network interface protocols of the data link layer Internet layer.
- RARP, the Reverse Address Resolution Protocol, translates hardware addresses to software addresses
- ICMP, the Internet Control Messaging Protocol, provides error-reporting mechanisms for regulating network performance
- SLIP, the Serial Line Internet Protocol, provides remote access
- CSLIP, the Compressed Serial Line Internet Protocol, provides remote access
- PPP, the Point-to-Point Protocol, provides remote access

3

Common TCP/IP Applications

Several TCP/IP applications provide support for network use. Many of these applications are based on RFC (Request for Comments) Internet Architecture Board (IAB) standards. Table 3–1 identifies commonly used network applications that are supported by almost all TCP/IP systems. The standard command, which is a UNIX syntax, is traditionally used with TCP/IP. Because this syntax can vary slightly among vendors, and because the syntax can be unfamiliar to OpenVMS users, an alternative syntax was developed jointly with the TCP/IP vendors for OpenVMS. This OpenVMS syntax is consistent with all TCP/IP vendor products and is a layer on OpenVMS commands. Note that the traditional UNIX style commands can also be used.

Table 3–1 Commonly Used TCP/IP Applications

Operation	Standard Name	Standard Command	OpenVMS DCL Command
Remote terminal service	RFC 854 and others,[1] Telnet	telnet	SET HOST/TELNET
	RFC 1282, BSD Rlogin	rlogin	SET HOST/RLOGIN
Remote file access	RFC 959, File Transfer Protocol (FTP)	ftp>get ftp>put	COPY/FTP
	BSD RCP[2]	rcp	COPY/RCP
Remote directory listings	RFC 959, File Transfer Protocol (FTP)	ftp>ls	DIR/FTP

[1] No single TELNET RFC exists. Consult vendor documentation for more information regarding supported RFCs.

[2] No RFC standard currently exists for RCP, the Berkeley file transfer protocol.

Users on OpenVMS systems running TCP/IP software can invoke certain TCP/IP applications by specifying related OpenVMS DCL commands (as indicated in Table 3–1). Chapter 5 specifies the DCL command formats.

TCP/IP applications follow a client/server model. The client is a program on a host that uses the services of a server located on either the same host or on a different host than the client. Among the most frequently encountered client/server user applications for TCP/IP are TELNET and FTP.

This chapter summarizes the characteristics of common TCP/IP applications. Software products supplied by Digital and other vendors implement these applications for OpenVMS systems. (See Appendix A for descriptions of currently available TCP/IP products that run as layered products on OpenVMS systems.)

This chapter also describes software tools for retrieving information on the Internet.

3.1 Remote Terminal Service

Virtual terminal protocol applications provide the TCP/IP application layer utilities and services that enable each user to log in to a remote host, using the local terminal as though it were a terminal on the remote host.

TCP/IP supports two virtual terminal protocols:

- TELNET, which is the RFC standard virtual terminal interface
- RLOGIN, which is the Berkeley standard virtual terminal protocol

Both the TELNET service and RLOGIN utility perform remote login operations, but in different ways.

3.1.1 Connecting to a Remote Server Using the TELNET Service

TELNET, running on a local computer, enables the user to connect to a specified host through a network that supports TCP/IP connections. TELNET can connect a computer to a remote computer host located anywhere in the world.

The TELNET service supports the **telnet** command, which allows a user on a client system to connect with a server on a remote system. The default server is the TELNET server that provides an interactive terminal session to execute commands at the remote host. The **telnet** command uses TCP as the transport protocol to guarantee accurate data delivery.

If the user connects to a TELNET server, a login session begins on the remote system. Once the connection is made, TELNET allows interaction between the user and the remote host. The user's keystrokes pass to the host and the remote computer displays pass to the local terminal. The local keyboard and screen seem to be connected to the remote computer.

In most cases, the user needs an account and password on the remote host in order to use TELNET for logging in to the remote system. Some software products that implement the **telnet** command also use Kerberos authentication to validate the user's identity (see Section 5.1.1) before granting access to the remote system. Many computers on the Internet provide some type of public TELNET access, permitting users to log in to a special guest account.

The **telnet** command models the local terminal and the remote terminal into network virtual terminals (simplified ASCII devices with keyboards and printers). Services required by either end of the connection are negotiated between the client and server.

TELNET also permits the local user to connect to an IBM mainframe host over a TCP/IP connection. A special TELNET client supports 3270 mode, which provides for IBM 3270 terminal emulation. When the TN3270 mode is active, the local keyboard emulates the keyboard normally used on an IBM 3270 class terminal. The remote IBM mainframe host must support the 3270 TELNET server.

In addition to the standard **telnet** command, OpenVMS users on systems running TCP/IP software can use the alternative OpenVMS style commands to make TELNET connections. For example, the DCL command SET HOST/TELNET invokes the TELNET client program (see Section 5.1.4). The DCL command SET HOST/TN3270 invokes the TN3270 terminal emulator client program (see Section 5.1.5).

3.1.2 Logging In to a Remote Host Using the RLOGIN Utility

The RLOGIN utility permits the user on a local host to run commands interactively on a remote host. The RLOGIN client program on the local host is connected to a remote RLOGIN server through a TCP/IP connection.

A user at a local host invokes the **rlogin** command. This command logs the user in to a remote host and starts an interactive terminal session. After the session starts, all commands the user enters at the local host are executed by the remote host, while the local host displays all of the output. The login session at the remote host can also be started with the name of a different user. If necessary, an 8-bit path can be established between the local and remote hosts.

The RLOGIN utility allows different hosts to share resources. When a host receives a request for connection, the RLOGIN server validates the source port number and the client user name. If the user validation fails, the RLOGIN server prompts for the user's password and checks the password file on the remote host. Some application products that implement the **rlogin** command use Kerberos authentication to validate the user's identity (see Section 5.1.1). The Kerberos authentication standard is RFC 1411.

In addition to the standard **rlogin** command, OpenVMS users on systems running TCP/IP software can use the alternative OpenVMS style commands to invoke RLOGIN programs. If you are logged in to an OpenVMS system that is running TCP/IP software, you can use the OpenVMS DCL command SET HOST/RLOGIN to invoke the RLOGIN client program (see Section 5.1.3).

3.2 Remote File Access

TCP/IP provides the following file access protocols:

- FTP, which is the RFC standard file transfer protocol
- RCP, which is the Berkeley file transfer protocol

Both FTP and RCP can copy files to and from remote hosts. In addition, FTP can manage remote directories. FTP services download or upload files over the Internet (see Section 3.3.1).

3.2.1 Transferring Files Between Hosts Using FTP

FTP (File Transfer Protocol) is a simple way to move files across a TCP/IP network. The **ftp** command invokes a utility that permits the user to transfer files between hosts that do not support the same file systems (for example, between UNIX and OpenVMS hosts). Normally, the user executing the ftp command must have a password on the remote host. Some systems use the " anonymous FTP " service, which accepts a *userid* of **anonymous** and no password. (The standard convention is to provide a complete electronic mail address for the password.)

The FTP utility permits the user to transfer groups of files between local and remote hosts. First, the user establishes an interactive connection with the FTP server on the remote host by providing a user name and password. Once the connection is established, the user can invoke ftp commands to transfer text and binary files to or from the remote host. Downloading of files from FTP sites on the Internet is described in Section 3.3.1.

In addition to the standard **ftp** command, OpenVMS users on systems running TCP/IP software can use the alternative OpenVMS style commands to invoke FTP services and perform copy operations. On OpenVMS systems running TCP/IP software, the DCL command COPY/FTP invokes the FTP service and performs a copy operation (see Section 5.2.3).

3.2.2 Listing Remote Host Directories Using FTP

The **ftp** command supports several file manipulation commands for managing remote directories. After establishing the FTP interactive session (as described in Section 3.2.1), the user can enter the ftp command **dir** or **ls**. This causes the remote system to list its directory contents on the local terminal. Other ftp commands provide directory operations such as creating, changing, or removing the remote directory and for other file operations over the TCP/IP network.

In addition to the standard **ftp**, **dir**, or **ls** commands, OpenVMS users on systems running TCP/IP software can use the alternative OpenVMS style commands to invoke FTP services that will list directories. On OpenVMS systems running TCP/IP software, the DCL command DIR/FTP invokes the FTP service and performs the directory listing operation (see Section 5.3.1).

3.2.3 Copying Files from Host to Host Using RCP

The RCP application consists of an RCP client utility running on the local host and an RCP server on each remote host involved in the copy operation.

The **rcp** command copies one or more files from one host to another or copies whole directory trees. Some implementations of the **rcp** command permit the local user to copy files from one remote host to another.

The RCP server validates the rcp command from the local host by checking whether the source host name is in the destination host's database, or by resolving the IP address in the Domain Name System. Some products that implement the **rcp** command also use Kerberos authentication (see Section 5.1.1).

In addition to the standard **rcp** command, OpenVMS users on systems running TCP/IP software can use the alternative OpenVMS style commands to invoke RCP services that will perform copy operations. OpenVMS users on systems running TCP/IP software can use the DCL command COPY/RCP to invoke the RCP utility (see Section 5.2.4).

3.3 Retrieving Information Through the Internet

The Internet offers many resources for obtaining information from systems on the thousands of networks that form the Internet. Internet software tools are designed to operate over TCP/IP connections. These tools use many of the same client/server techniques as the common FTP and TELNET applications. In many instances, the tools for retrieving information are FTP and TELNET.

3.3.1 Downloading Files from FTP Sites on the Internet

FTP provides an important service to Internet users by allowing them to move large quantities of data across the Internet. FTP sites on the Internet provide huge data storage facilities, storing files of all types. You can browse through the names and descriptions of files at FTP sites and download the files to your computer. Many files found on an FTP server are stored in a compressed format. When you receive a file, you can decompress (expand) the file to its original size.

Many sites permit you to log in as an anonymous user, which means you do not need a password. Anonymous FTP allows an organization to distribute certain files (for example, electronic magazines) for free to the general public.

3.3.2 Using Browsers with the World Wide Web (WWW)

The World Wide Web (WWW) is a client/server hypermedia system that runs over the Internet. Developed by CERN in Switzerland, [1] the WWW is an information management facility adopted by thousands of sites worldwide. Users access the WWW by pointing their client software at any of thousands of server connections. Hypertext multimedia documents stored in the WWW contain links to other data. The hypertext format allows you to retrieve and display data based on keyword searches.

Software tools called browsers are available to access data on the World Wide Web. The Mosaic browser is a hypertext interface to the WWW that has links to files, images, text, audio, and video. Mosaic, which is a graphical single-client application, offers point and click menus.

When you run Mosaic and connect to the WWW, the first display is usually a "home page." The home page is specific to the site you selected and often contains a welcome message and hyperlink indicators. These indicators are highlighted phrases or graphics, which may be in color or underlined or both, that link to more specific information. By clicking on the hyperlinks you can access data anywhere on the web.

For character-cell terminals, you can use the Lynx browser to interface with the WWW. Keyword links are highlighted on the screen. You can advance the cursor to a keyword and jump to the linked document residing anywhere on the web.

Mosaic is available on the Motif kit. WWW tools and a WWW server are available on the Freeware CD.

3.3.3 Using the Gopher Service to Access Internet Resources

The Gopher service is a client/server system that provides a simple, consistent means of accessing the full resources of the Internet. The Gopher is a highly automated software directory.

The Gopher client displays a series of menus from which you can make selections that the client carries out (for example, getting a file from a remote computer). Gopher menu items can lead to other menus at different sites.

Gopher servers are set up at many companies, universities, and other organizations to provide information of interest to local users. Many of the Gopher servers are public and supply information of interest to the general public. Gopher servers in many countries throughout the world are interconnected to form the Internet Gopher.

A Gopher client and server are available on the Freeware CD.

---

[1] For more information, open the following uniform resource locator (URL) on the WWW:

`http://info.cern.ch/hypertext/WWW/TheProject.html`

3.3.4 Sending Electronic Mail over the Internet

Once you are connected to the Internet, you can send electronic mail (or e-mail) messages over TCP/IP connections to systems throughout the Internet. You can also send mail to accounts that do not use Internet addressing; gateways translate the messages and send them to the appropriate networks and systems.

To send mail to an Internet address, use the format *user@node.org* according to the guidelines in the following list.

- If the Internet transport on your machine is not SMTP, define the logical name MAIL$INTERNET_TRANSPORT to select an alternate transport.
- If you specify a DECnet phase IV node or a DECnet/OSI alias, you may use the *user@node* syntax. In this case, the syntax is treated as NODE::USER.

The format is also available in character-cell mail and DECwindows mail. For more information about the format for specifying Internet mailing addresses see the *OpenVMS User's Manual*.

Internet host computers may also maintain lists called "mailing lists," which are databases of people who have shared interests in a topic. Sending e-mail to the mailing list causes the mail to be sent to everyone on the list. Examples of messages include articles, comments, and other information about the topic of the mailing list. You can get on a mailing list by sending e-mail to the mailing list administrator. Some mailing lists have moderators who screen the messages for duplication and inappropriate content.

3.3.5 Using UseNet to Access Internet Newsgroups

The UseNet protocol describes how to store and send groups of messages between computers that may or may not be on the Internet. UseNet (the User's Network) is a virtual forum divided into newsgroups that deal with various topics. The site administrator who sets up a newsgroup determines the topic of the newsgroup. Internet newsgroups are similar to notes conferences; they provide vast amounts of information.

You can use a UseNet reader software program to access a newsgroup. Discussions are conducted by sending e-mail messages to the newsgroup's address. If the newsgroup is moderated, the moderator screens the mail before posting it to the newsgroup.

A news reader, mxrn, is available on the Freeware CD.

4

Mapping UNIX to OpenVMS Identification Code

TCP/IP networking applications support general user operations that access resources, such as files. One method of controlling who can access operating system resources is by assigning identification codes to users.

Both OpenVMS based systems and UNIX-based systems use identification codes as a general method of resource protection. However, each of the operating systems implement the coding differently. Because the TCP/IP software was originally developed on and used for UNIX machines, TCP/IP implementations use UNIX-style identification codes. Consequently, some TCP/IP applications must map UNIX identification codes to OpenVMS identification codes.

This chapter includes summary information about OpenVMS and UNIX identification codes and the mapping mechanisms used by TCP/IP applications.

4.1 What are UIDs and GIDs?

OpenVMS users are familiar with the OpenVMS user identification code (UIC) that identifies the user as a member of a group that can share specific data. The UIC corresponds to the name of the user who created the process running on OpenVMS. The UIC is a 32-bit field comprising a 14-bit user number and a 14-bit group number. UIC-based protection controls access to such objects as files and directories.

Just as OpenVMS employs user names and UICs for identification, UNIX identifies users by user names and a user identification (UID) group identification (GID) pair. Both UIDs and GIDs are simply numbers that can identify a user on a system. Some versions of UNIX (for example, DEC OSF/1) use 32-bit UID/GID pairs.

Some TCP/IP applications require use of UID/GID pairs for user identification. The most common application that requires this identification is the NFS (Network File System) client/server application. To use this application on OpenVMS, you must map OpenVMS user names to UNIX-style GID/UID pairs.

4.2 Establishing the Relationship Between UID/GID Pairs and OpenVMS User Names

All TCP/IP vendors for OpenVMS support mechanisms for mapping OpenVMS user names to UID/GID pairs. Consult the appropriate vendor documentation for more information about how to manage this process for a particular OpenVMS TCP/IP layered product. (Appendix A lists TCP/IP layered products that run on OpenVMS.)

5

Commands Common to All TCP/IP Products That Run on OpenVMS

OpenVMS users can use familiar DCL commands, with special parameters and qualifiers, to perform general user functions involving remote systems accessible over TCP/IP networking connections. In addition to the standard UNIX style syntax, the alternative OpenVMS style syntax is available, and is described in this chapter.

DCL commands that support TCP/IP parameters and qualifiers include the COPY, DIRECTORY, and SET HOST commands. DCL command formats enable you to log in, connect to a remote host over a TCP/IP connection, copy or transfer files from host to host, and display remote host directories on the local host. The local host and any remote hosts involved in these operations must include software that supports TCP/IP protocols.

Appendix A describes layered TCP/IP software products that run on OpenVMS and the vendors that supply the standard commands for accessing the TCP/IP applications.

5.1 Virtual Terminal Services

If you are a user on an OpenVMS client system running TCP/IP software, you can use SET HOST commands to access virtual terminal services, including:

- Logging in to a remote system using the RLOGIN utility
- Connecting to a remote system using the TELNET service, including connecting to an IBM system using the TN3270 terminal emulator

This section presents the formats of the SET HOST commands supported over TCP/IP connections:

- SET HOST/RLOGIN
- SET HOST/TELNET
- SET HOST/TN3270

The SET HOST commands invoke the RLOGIN and TELNET client programs. A remote host is identified either by its Internet Protocol (IP) host name or by its IP address (see Chapter 2). See Section 3.1 for a description of the TCP/IP client/server applications that supply remote terminal services.

5.1.1 Kerberos Authentication

Kerberos authentication is a system that verifies the identity of users, thereby providing security in a network environment that otherwise may not be secure. Some TCP/IP software products use Kerberos to make sure the identity of any user who requests access to a remote host is authentic.

Kerberos keeps a control list of users and their encrypted passwords. Users must obtain Kerberos **tickets** to invoke utilities that support commands with special Kerberos enhancements. RCP, RLOGIN, and TELNET applications can support Kerberos enhancements.

When a TCP/IP layered product supports Kerberos authentication, a user on an OpenVMS system can specify the /AUTHENTICATE qualifier in any of the following OpenVMS DCL commands:

SET HOST/RLOGIN
SET HOST/TELNET
SET HOST/TN3270
COPY/RCP

5.1.2 Case-Sensitive Forms of /USERNAME Value

The DCL commands SET HOST/RLOGIN and COPY/RCP support the following three distinct case-sensitive forms of the /USERNAME qualifier value. Enclose the user name in quotes to preserve its case.

1. Normally, the *username* is all lowercase characters:

```
$ set host/rlogin/username=ralf host.dom
```

2. The *username* can contain mixed-case characters:

```
$ set host/rlogin/username="RaLf" host.dom
```

3. The *username* can contain all uppercase characters:

```
$ set host/rlogin/username="RALF" host.dom
```

5.1.3 SET HOST/RLOGIN

This command logs the user in to a remote host over a TCP/IP connection and starts an interactive terminal session by accessing the RLOGIN application.

Format

$$\text{SET HOST/RLOGIN} \left\{ \begin{array}{l} \text{IPhostname} \\ \text{IPaddress} \end{array} \right\}$$

Parameters

IPhostname
Specifies the IP host name of the remote host.

IPaddress
Specifies the IP address of the remote host.

Description

The SET HOST/RLOGIN command allows you to log in to a remote host. The command invokes RLOGIN client software that runs on your system. After the terminal session starts, you can enter commands interactively on the remote host. The parameter that specifies the remote host is either an IP host name or an IP address. The /RLOGIN qualifier is a required part of the format. Other available qualifiers are in the following list.

Qualifiers

/AUTHENTICATE
Specifies that Kerberos authentication should be used for acquiring access to the remote host.

/PASSWORD=password
Optional qualifier that specifies the password of the user who is logging in to the remote host.

/TERMINAL_TYPE=type
Sets the terminal emulator to one of the following terminal types: VT100, VT200, VT300, VT400, VT500.

/TRUNCATE_USERNAME
Specifies that the current user name must be truncated to 8 characters before attempting to connect to the remote host. The qualifier is required for communication with systems that limit the size of their login names to 8 characters. The /TRUNCATE_USERNAME qualifier is ignored if /USERNAME is specified.

/USERNAME=username
Specifies the user name for logging in to the remote host. The user name can be enclosed in quotes to preserve the case of the user name for case-sensitive systems such as UNIX (see Section 5.1.2). If the /USERNAME qualifier is not specified, the default is the current user's name.

Example

```
DCL> SET HOST/RLOGIN REMOTEHST1
```

This example creates an RLOGIN connection to remote host REMOTEHST1 over a TCP/IP connection.

5.1.4 SET HOST/TELNET

This command connects you to a remote host over a TCP/IP connection by invoking the TELNET application.

Format

$$\text{SET HOST/TELNET} \begin{Bmatrix} \text{IPhostname} \\ \text{IPaddress} \end{Bmatrix}$$

Parameters

IPhostname
Specifies the name of a remote IP host.

IPaddress
Specifies the IP address.

Description

The SET HOST/TELNET command allows you to connect to a server on a remote system by invoking the TELNET client software that runs on your system. The parameter that specifies the remote host is either an IP host name or an IP address. Most of the attributes are negotiated with the remote host. The qualifiers are used only in exception cases (for example, cases where a remote server does not support a negotiated parameter but requires a certain characteristic for the connection). The /TELNET qualifier is required in the format. Other available qualifiers are in the following list.

Qualifiers

/AUTHENTICATE
Specifies that Kerberos authentication should be used for acquiring access to the remote host.

/PORT=port
Specifies the remote TCP port to use. The default is 23.

/TERMINAL_TYPE=type
Sets the terminal emulator to one of the following terminal types: VT100, VT200, VT300, VT400, VT500.

Example

```
DCL> SET HOST/TELNET remotehst2
```

This example creates a TELNET connection to remote host *remotehst2* over a TCP/IP connection.

5.1.5 SET HOST/TN3270

This command connects your local host to a remote IBM host over a TCP/IP connection and invokes the TN3270 terminal emulator TELNET client program.

Format

$$\text{SET HOST/TN3270} \left\{ \begin{array}{l} \text{IPhostname} \\ \text{IPaddress} \end{array} \right\}$$

Parameters

IPhostname
Specifies the name of an IP host.

IPaddress
Specifies an IP address.

Description

The SET HOST/TN3270 command makes a connection with a TELNET server on a remote IBM system and invokes the TN3270 terminal emulator TELNET client program on the local host. The parameter that specifies the remote host is either an IP host name or an IP address. The TN3270 client automatically determines the terminal type (IBM-3278-2, IBM-3278-3, IBM-3278-4, or IBM-32798-5). The /TN3270 qualifier, which is required in the format, causes the local keyboard to emulate an IBM 3279 class terminal keyboard by invoking the TN3270 terminal emulator. Other available qualifiers are in the following list.

Qualifiers

/AUTHENTICATE
Specifies that Kerberos authentication should be used for acquiring access to the remote host.

/PORT=port
Specifies the remote TCP port to use. The default is 23.

Example

```
DCL> SET HOST/TN3270 REMOTEHST3
```

This example creates a connection to a TELNET server on the remote IBM system REMOTEHST3 over a TCP/IP connection.

5.2 File Transactions

OpenVMS DCL commands support file manipulation utilities over TCP/IP connections, including:

- FTP, which transfers files between hosts
- RCP, which permits files to be copied from host to host

The file access applications require a utility at the local terminal and one or more remote servers. See Section 3.2 for a description of remote file access utilities supported by TCP/IP.

5.2.1 File Length and File Format

The majority of files copied are ASCII text or binary images. These files are handled properly by all TCP/IP vendors' RCP or FTP applications. TCP/IP was written for UNIX systems, which use 512-byte blocks. OpenVMS, however, uses Record Management Services (RMS) as the native file system. RMS handles variable-length records and multiple file formats. Copying a file with any kind of record-oriented format to a UNIX system causes the file attributes to be lost.

Some of the TCP/IP products that run on OpenVMS (as described in Appendix A), allow copying of an FDL (file definition language) file so that OpenVMS can restore RMS file attributes retrieval. Most TCP/IP products on OpenVMS support special copying modes that preserve file attributes when files are copied to or from OpenVMS systems.

5.2.2 Remote File Specification Format

You can use OpenVMS DCL commands to access remote files over TCP/IP connections simply by including in the file specification the name of the remote host on which the file is located. You can access files that are protected against general access if the owner has provided you with the name and password of the account.

The OpenVMS DCL commands for TCP/IP support the same remote file specification format as the DCL commands for DECnet network connections. Some implementations of the file transaction applications support file transfers in which both the source file and the destination file are remote file specifications.

The full format for a remote file specification is as follows:

host"username password account"::remote-file-spec

If a file resides on a system other OpenVMS, enclose the name of the file in quotation marks. For example, to access a file named */usr/users/user/Orders* on a DEC OSF/1 host named U32, you would use the following format for the file specification:

```
U32"user password"::"/usr/users/user/Orders"
```

Unlike OpenVMS, UNIX systems support case-sensitive file specifications.

(See also Section 5.1.2 for a description of case-sensitive user names.)

5.2.3 COPY/FTP

This command transfers files between hosts, that possibly have dissimilar file systems, over a TCP/IP connection by invoking the FTP utility.

Format

COPY/FTP from-file to-file

Parameters

from-file
Specifies the name of an existing file (the source file) to be copied.

to-file
Specifies the name of the output file (the destination file) into which the input file is copied.

Description

The COPY/FTP command copies files to and from remote hosts using the File Transfer Protocol (FTP). The services provided by this command are a subset of the architected features of FTP. (See the vendor documentation for usage of their supplied FTP program.) The /FTP qualifier is required in the format. Other available qualifiers are in the following list.

Note

In OpenVMS to OpenVMS transfers, if both machines support VMS structured transfers, the /BINARY, /ASCII, and /FDL qualifiers are ignored. The cooperating OpenVMS FTP client and server automatically transfer the file with proper OpenVMS attributes.

Qualifiers

/ANONYMOUS
Causes an anonymous access to the remote host or hosts. /ANONYMOUS is the default remote access. The password passed to the remote host must have the form *user@fully-qualified-host-name*.

/ASCII
Identifies an ASCII text file. /ASCII is the default.

/BINARY
Identifies binary files.

/FDL
Causes interaction with an FDL (file definition language) file. If the file is being copied to the local OpenVMS system, a remote FDL file is sought and interpreted for the operation. If the file is being copied outside the local OpenVMS system, an FDL file is generated and copied in addition to the requested file. If the /FDL qualifier is specified and the vendor application does not support it, a warning message may be issued. This qualifier is optional.

/LOG
Displays a message at SYS$OUTPUT when a file is transferred.

/NOSTRUVMS
Explicitly disables the negotiation of STRUVMS transfers. Otherwise, some servers will immediately abort when negotiating the feature.

/VERBOSE
/NOVERBOSE
Specifies whether all messages (including banner messages) are to be displayed on the terminal. By default, disables the display of the messages.

Examples

```
DCL> COPY/FTP/ASCII/ANON ovms_file1.c remotehst5::"/public/ovms_file2.c"
```

This example transfers the local text file OVMS_FILE.C to the remote file /public/ovms_file2.c on remotehst5 using anonymous access over a TCP/IP connection.

```
DCL> COPY/FTP/FDL/ANON rms_indexed_file.idx remotehst5::"/public/rms.idx.file"
```

This example transfers the OpenVMS RMS file RMS_INDEXED_FILE.IDX to the remote file PUBLIC/RMS.IDX.FILE on REMOTEHST5 over a TCP/IP connection. The access to the remote host is anonymous; an FDL file is generated and copied along with RMS_INDEXED_FILE.IDX.

5.2.4 COPY/RCP

This command copies files from host to host over a TCP/IP connection by invoking the RCP utility.

Format

COPY/RCP from-file to-file

Parameters

from-file
Specifies the name of an existing file (the source file) to be copied.

to-file
Specifies the name of the output file (the destination file) into which the input file is copied.

Description

The COPY/RCP command copies one or more files (or directory trees) to or from a remote host using the RCP utility.

The file specification format is described in Section 5.2.2.

The /RCP qualifier, which automatically recognizes binary and ASCII formats, is required in the format. Other available qualifiers are in the following list.

Qualifiers

/LOG
Displays a message at SYS$OUTPUT when a file is transferred.

/PRESERVE
Preserves the file protection codes.

/RECURSIVE
Requests a subdirectory copy operation.

/TRUNCATE=USERNAME
Truncates the user name to 8 characters.

/USERNAME=username
Optional qualifier that specifies the remote user name. The standard operation is to log in to a remote system using the same user name as at the local terminal. The command supports quoted parameters in the /USERNAME value (see Section 5.1.2).

Example

```
DCL> COPY/RCP local_file.c remotehst4"Smith smpw"::rem_file.c
```

This example copies LOCAL_FILE.C to REM_FILE.C on the remote host REMOTEHST4 over a TCP/IP connection.

5.3 Directory Transactions

FTP applications support commands to manage the directory to or from which you are transferring files. The DCL command DIR/FTP gets a list of the contents of the remote directory. Section 3.2.2 describes how to use FTP in listing remote directories.

5.3.1 DIR/FTP

This command displays remote directory information over a TCP/IP connection by invoking the FTP utility.

Format

DIR/FTP directory-spec

Parameters

directory-spec
Specifies the standard DECnet remote file specification. A quoted file string preserves the case (for case-sensitive systems such as UNIX) and identifies a foreign device and directory specification (see Section 5.2.2).

Description

The DIR/FTP command writes a listing of the contents of the specified remote directory to the local host over a TCP/IP connection by invoking the FTP utility. The /FTP qualifier is required in the format. Other available qualifiers are in the following list.

Qualifiers

/ANONYMOUS
Causes an anonymous access to the remote host or hosts. /ANONYMOUS is the default remote access. The password passed to the remote host must have the form *user@fully-qualified-host-name*.

/FULL
Displays additional information about the files in the specified remote directory if the /FULL qualifier is supported by the remote server.

Example

```
DCL> DIR/FTP remotehst6"Jones jpw"::"usr/public"
```

This example causes the contents of the remote directory USR/PUBLIC on remote host REMOTEHST6 to be displayed at the local host over a TCP/IP connection.

A

OpenVMS TCP/IP Software Vendors

The following list identifies vendors and the TCP/IP products they supply. These products run on OpenVMS systems as layered software.

- Digital Equipment Corporation, DEC TCP/IP Services for OpenVMS

  Digital
  DEC TCP/IP Service for OpenVMS Product Manager, LKG2A/T12
  550 King St.
  Littleton, MA 01460
  Phone:
  (800) DIGITAL
  (508) 486-5095
  FAX:
  (508) 486-7417
  (Attention: DEC TCP/IP Service for OpenVMS Product Manager, LKG2A/T12)
  World Wide Web:
  http://www.Digital.com/

- Process Software Corporation, TCPware for OpenVMS

  959 Concord Street
  Framingham, MA 01701
  Phone:
  (800) 722-7770
  (508) 879-6994
  FAX:
  (508) 879-0042
  Internet E-mail:
  info@process.com
  World Wide Web:
  http://www.process.com/

- TGV, Inc., MultiNet

  101 Cooper Street
  Santa Cruz, CA 95060
  Phone:
  (800) TGV-3440
  (408) 457-5200
  FAX:
  (408) 457-5205
  Internet E-mail:
  sales@tgv.com
  World Wide Web:
  http://www.tgv.com/

- The Wollongong Group, Inc., PathWay

```
1129 San Antonio Rd.
PO Box 51860
Palo Alto, CA  94303
   Phone:
     (800) 872-8649 (outside of CA)
     (415) 962-7202 (in CA)
   FAX:
     (415) 962-0286
   Internet E-mail:
     sales@twg.com
   World Wide Web:
     http://www.twg.com/
```

B

Using DECnet Over TCP/IP

DECnet/OSI for OpenVMS enables you to run Digital Network Architecture (DNA) and OSI applications over an IP network backbone, as well as over OSI or traditional DECnet backbones. Applications include those supplied by Digital, third-party applications, and user-written applications. For details on configuring DECnet over TCP/IP, see the *DECnet/OSI for OpenVMS Installation and Configuration* guide.

When to Use TCP/IP or DECnet Over TCP/IP

Use TCP/IP in the following situations:

- For access to nodes not running DECnet/OSI
- For TCP applications such as:

  SET HOST/TELNET
  SET HOST/RLOGIN
  COPY/FTP
  COPY/RCP
  DIRECTORY/FTP
  telnet
  rlogin
  ftp
  rcp

Use DECnet over TCP/IP in the following situations:

- For traditional DECnet applications such as:

  SET HOST
  MAIL
  DIR

- For applications written to the NET $QIO or $IPC interface
- For OpenVMS connection auditing and proxy access

B.1 Establishing Network Connections

You can use DECnet over TCP/IP to connect to any of the remote hosts on the network. The following examples make the necessary connections.

To connect from a DECnet/OSI node to a DECnet-only node, use the following format:

```
$ SET HOST DECnet-only-node
```

To connect from a DECnet/OSI node to another DECnet/OSI node, use the following format:

```
$ SET HOST DECnet-OSI-node.computer-name.company.com
```

To connect from a DECnet/OSI node to a TCP/IP node, use either of the following formats:

```
$ SET HOST/TELNET TCP-IP-node.computer-name.company.com
```

```
$ SET HOST/RLOGIN TCP-IP-node.computer-name.company.com
```

B.2 Using DECnet Applications RFC1006 and RFC1006 Plus

RFC1006 is an Internet standard that defines how to implement ISO 8073 Class 0 on top of TCP. Hosts that implement RFC1006 must listen on TCP port 102.

RFC1006 Plus is an Internet draft that defines how to implement ISO8073 Transport Class 2 Non-Use of Explicit Flow Control on top of TCP. Hosts that implement RFC1006 Plus must listen on TCP port 399.

The DECnet over TCP/IP feature, which is RFC1006 Plus, allows traditional DECnet applications (such as MAIL, CTERM, and FAL) to accept IP names and addresses. The OSI applications over TCP/IP feature, which is RFC1006, allows OSI applications (such as FTAM and VTP) to accept IP names and addresses. An IP name specification is translated from the BIND server(s) defined in the local TCP/IP product already installed on your system.

Index

AA-QJGDA-TE

Printed in Europe

Printed on Environmentall
Friendly Paper